

FastAPI+LlamaIndex

以下是一个结合 FastAPI 和 LlamaIndex 实现 RAG（检索增强生成）功能的案例，将各个环节串一下，备查。

环境准备

系统环境

使用 modelscope，30 个小时的免费使用，注册就行
<https://www.modelscope.cn/my/mynotebook>



Conda 环境

AI 开发中使用的各工具版本大多是 0.X 版本、技术迭代快、工具也相互依赖，开发中最让痛苦的就是平衡依赖的版本，让程序“当前”能正常运行。所以，需要隔离 python 环境就行，

安装软件

下载

wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh

```
root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace# wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
—2025-03-08 11:40:51— https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
正在解析主机 repo.anaconda.com (repo.anaconda.com)... 104.16.32.241, 104.16.191.158, 2606:4700::6810:20f1, ...
正在连接 repo.anaconda.com (repo.anaconda.com) | 104.16.32.241 | :443... 已连接。
```

安装

bash Miniconda3-latest-Linux-x86_64.sh

```
root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace# bash Miniconda3-latest-Linux-x86_64.sh

Welcome to Miniconda3 py312_24.11.1-0

In order to continue the installation process, please review the license
agreement.
Please, press ENTER to continue
>>>
ANACONDA TERMS OF SERVICE
```

各种ok下一步

激活

source ~/.bashrc

```
modified      /root/.bashrc

==> For changes to take effect, close and re-open your current shell. <==

Thank you for installing Miniconda3!
root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace# source ~/.bashrc
(base) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace#
```

验证

conda --version

```
(base) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace# conda --version
conda 24.11.1
```

创建环境

创建

conda create -n myenv python=3.10

```
(base) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace# conda create -n myenv python=3.10
Channels:
- defaults
Platform: linux-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##
```

注意：python3.10 是“目前”兼容性最好的版本，无特别的依赖要求就用该版本。

激活

conda activate myenv

```
(base) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace# conda activate myenv
(myenv) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace#
```

注：后面的操作都是在 myenv 环境中操作

其他

[pip install llama-index](#)

```
(myenv) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace# pip install llama-index
Looking in indexes: https://mirrors.cloud.aliyuncs.com/pypi/simple
Collecting llama-index
  Using cached https://mirrors.cloud.aliyuncs.com/pypi/packages/ba/9f/5998f4b2e21e9b78dbcf4dbbdaa73216898b15a72d3ad
py3-none-any.whl (7.0 kB)
Collecting llama-index-agent-openai<0.5.0,>=0.4.0 (from llama-index)
  Downloading https://mirrors.cloud.aliyuncs.com/pypi/packages/8a/1f/a0e2eed0417b1f3b6a51da159eb57640f0501e74fd502f
nai-0.4.6-py3-none-any.whl (13 kB)
Collecting llama-index-cli<0.5.0,>=0.4.1 (from llama-index)
  Downloading https://mirrors.cloud.aliyuncs.com/pypi/packages/ae/fa/2ee58764d733e9b5d61036ba6c8c96adcd567ea16a62c
-py3-none-any.whl (28 kB)
Collecting llama-index-core<0.13.0,>=0.12.23 (from llama-index)
```

[pip install llama_index.vector_stores.chroma](#)

```
(myenv) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace/public/fastapi# pip install llama_index.vector_stores.chroma
Looking in indexes: https://mirrors.cloud.aliyuncs.com/pypi/simple
Collecting llama_index.vector_stores.chroma
  Downloading https://mirrors.cloud.aliyuncs.com/pypi/packages/77/24/01f707c6a61887d8011df68c4b4ff6d96d985955e86a0b8638ef98
ores_chroma-0.4.1-py3-none-any.whl (5.2 kB)
Collecting chromadb>=0.5.17 (from llama_index.vector_stores.chroma)
  Downloading https://mirrors.cloud.aliyuncs.com/pypi/packages/28/8e/5c186c77bf749b6fe0528385e507e463f1667543328d76fd00a49e
ne-any.whl (611 kB)
 611.1/611.1 kB 10.5 MB/s eta 0:00:00
```

RAG 构建

从流程上梳理，一般 AI 项目的方案，数据>方案>模型>训练>评测>部署，这里只列出关键环节。

数据

重点，数据来源、数据结构、数据格式化问题，网络协议相关的数据集，准备好后，上传至服务器

```
protocol.txt X
public > fastapi > code > sample_docs > protocol.txt
36 3.4 广播
37 定义地址只是第一步，后面还有更多的步骤。
38 首先，一块网卡怎么会知道另一块网卡的MAC地址？
39 回答是有一种ARP协议，可以解决这个问题。这个留到后面介绍，这里只需要知道，以太网数据包的
40 其次，就算有了MAC地址，系统怎样才能把数据包准确送到接收方？
41 回答是以太网采用了一种很"原始"的方式，它不是把数据包准确送到接收方，而是向本网络内所
42
43 上图中，1号计算机向2号计算机发送一个数据包，同一个子网络的3号、4号、5号计算机都会收到
44 有了数据包的定义、网卡的MAC地址、广播的发送方式，"链接层"就可以在多台计算机之间传送数据
45 四、网络层
46 4.1 网络层的由来
47 以太网协议，依靠MAC地址发送数据。理论上，单单依靠MAC地址，上海的网卡就可以找到洛杉矶的网卡
```

模型

检索模型

检索模型一般选大一点，RAG 检索需基于模型本身智能，所以模型要越大效果越好；

ModelScope

首页模型库数据集创空间AIGC专区文档中心社区GitHub

通义千问1.5-7B-Chat

Qwen / Qwen1.5-7B-Chat

文本生成TransformersSafetensorsPyTorchqwen2开源协议: other英语chatPAI Model Gallery DeployPAI Model Gallery Train

@通义千问 提供 | 520,446下载 | 2025-02-26更新

模型介绍模型文件交流反馈19

当前模型介绍使用的语言，与您偏好语言不同，是否查看平台翻译的模型介绍？查看翻译版本


Qwen1.5-7B-Chat

Introduction

Qwen1.5 is the beta version of Qwen2, a transformer-based decoder-only language model pretrained on a large amount of data. In comparison with the previous version, the improvements include:

词嵌入模型

常用的一个 <https://www.modelscope.cn/models/BAAI/bge-small-zh-v1.5/summary>

 **bge-small-zh-v1.5**

BAAI / bge-small-zh-v1.5

特征抽取

PyTorch

Transformers

Safetensors

bert

开源协议: mit

中文

@北京智源人工智能研究院 提供 | 11,809下载 | 2024-11-21更新

模型介绍

模型文件

交流反馈 1

当前模型介绍使用的语言，与您偏好语言不同，是否查看平台翻译的模型介绍？ [查看翻译版本](#)

FlagEmbedding

[Model List](#) | [FAQ](#) | [Usage](#) | [Evaluation](#) | [Train](#) | [Contact](#) | [Citation](#) | [License](#)

More details please refer to our Github: [FlagEmbedding](#).

[English](#) | [中文](#)

FlagEmbedding can map any text to a low-dimensional dense vector which can be used for tasks like retrieval, classification, clustering, or semantic search. And it also can be used in vector databases for LLMs.

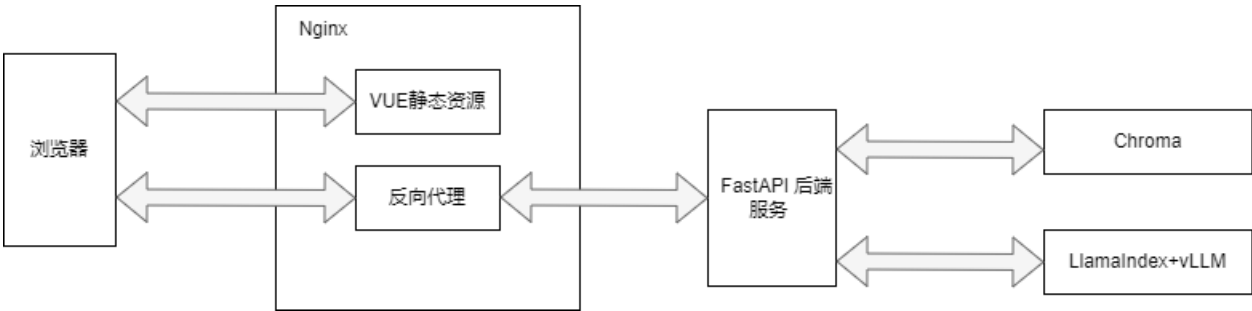
部署

总体

工具表

组件	技术	功能
后端	FastAPI	REST API、文档处理、RAG 核心逻辑
前端	Vue3 + Element Plus	用户界面、文档上传、问答交互
AI 框架	LlamaIndex	文档分块、向量化、检索增强生成
向量数据库	Chroma	文档向量存储与相似性搜索

结构图



服务	端口	服务名	功能说明
Nginx	80/443	gateway	API 网关, 入口代理、SSL、静态资源
Vue	80	frontend	前端静态资源 (Nginx 托管)
FastAPI/LlamaIndex	8000	rag-api	处理用户请求和调用 RAG 流程
Chroma	8000	chroma	向量数据库服务

后端

环境准备：

使用框架

- 使用 FastAPI+LlamaIndex
FastAPI: <https://fastapi.tiangolo.com/zh/>
LlamaIndex: <https://docs.llamaindex.ai/en/stable/>

- 安装依赖

pip install fastapi uvicorn llama-index python-dotenv pydantic sentence-transformers

```
(myenv) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace/public# C
(myenv) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace/public# ^C
(myenv) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace/public# curl -X POST "http://localhost:8000/query" -H "Content-Type: application/json" -d '{"qu
estion": "什么是路由?"}'
{"detail": "got an unexpected keyword argument 'max_length'"} (myenv) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace/public#
```

目录结构：

```
(myenv) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace/public/fastapi/code# tree
.
├── app.py
├── sample_docs
│   └── protocol.txt
```

编码 app.py

功能设计

文件名: app.py

功能设计：

- /query: 处理自然语言查询
- /health: 提供健康检查
- 返回结果包含: 答案、来源文档片段

关键实现解析

- **模型配置：**
 - 使用 BAAI/bge-small-zh-v1.5 作为本地嵌入模型
 - 使用 Qwen1__5-1__8B-Chat 作为本地生成模型
 - 可通过修改常量切换为 OpenAI 等云服务
- **索引构建：**
 - 自动监控 sample_docs 目录变化
 - 支持多文档格式（PDF/TXT/MD 等）
 - 使用 HuggingFace 模型进行本地向量化
- **API 设计：**
 - /query：处理自然语言查询
 - /health：提供健康检查
 - 返回结果包含：答案、来源文档片段
- **错误处理：**
 - 自动检测文档目录是否存在
 - 捕获模型加载和查询过程中的异常
 - 返回标准化的 HTTP 错误代码

功能编码

```
app.py  X
public > fastapi > code > app.py > ...
1  """
2  基于FastAPI和LlamaIndex的RAG服务
3  实现功能：文档加载、索引构建、自然语言问答
4  """
5
6  import os
7  from typing import Optional
8  from fastapi import FastAPI, HTTPException
9  from pydantic import BaseModel
10 from llama_index.core import VectorStoreIndex, SimpleDirectoryReader, Settings
11 from llama_index.core.embeddings import BaseEmbedding
12 from llama_index.embeddings.huggingface import HuggingFaceEmbedding
13 from llama_index.llms.huggingface import HuggingFaceLLM
14 from dotenv import load_dotenv
15 from fastapi.middleware.cors import CORSMiddleware  # CORS
16
17 # 加载环境变量（如需使用OpenAI等云服务）
18 load_dotenv()
19 # -----
20 # 配置参数（按需修改）
21 # -----
22 DOC_DIR = "./sample_docs"  # 文档存储目录
23 EMBED_MODEL = r"/mnt/workspace/llm/BAAI/bge-small-zh-v1.5"  # 本地嵌入模型
24 LLM_MODEL = r"/mnt/workspace/llm/Qwen/Qwen1.5-1.8B-Chat"  # 本地LLM模型
25
26 # -----
27 # FastAPI应用初始化
28 # -----
29 app = FastAPI(
30     title="RAG API Service",
31     description="基于本地模型的检索增强生成系统",
32     version="1.0"
33 )
34 # 添加 CORS 配置
```



```
36 # -----
37 # LlamaIndex组件初始化
38 # -----
39 class RAGSystem:
40     def __init__(self):
41         self.index = None
42         self.query_engine = None
43         self._init_models()
44         self._load_data()
45
46     def _init_models(self):
47         """初始化嵌入模型和LLM"""
48         # 使用本地嵌入模型
49         self.embed_model = HuggingFaceEmbedding(model_name=EMBED_MODEL)
50
51         # 配置本地LLM
52         self.llm = HuggingFaceLLM(
53             model_name=LLM_MODEL,
54             tokenizer_name=LLM_MODEL,
55             device_map="auto", # 自动选择GPU/CPU
56             generate_kwargs={"temperature": 0.1, "max_length": 500} # 此处统计配置生成长度
57         )
58
59         # 全局设置
60         Settings.embed_model = self.embed_model
61         Settings.llm = self.llm
```

```

63     def _load_data(self):
64         """加载文档并构建索引"""
65         try:
66             if not os.path.exists(DOC_DIR):
67                 os.makedirs(DOC_DIR)
68                 raise FileNotFoundError(f"请将文档放入{DOC_DIR}目录")
69
70             # 读取文档 (支持pdf、txt、md等格式)
71             documents = SimpleDirectoryReader(DOC_DIR).load_data()
72
73             # 构建向量索引
74             self.index = VectorStoreIndex.from_documents(documents)
75
76             # 创建查询引擎
77             self.query_engine = self.index.as_query_engine(
78                 similarity_top_k=3, # 检索前3个相关段落
79                 response_mode="compact" # 生成紧凑回答
80             )
81         except Exception as e:
82             raise RuntimeError(f"索引初始化失败: {str(e)}")
83
84     # 初始化RAG系统
85     rag_system = RAGSystem()
86
87     # -----
88     # API数据模型
89     # -----
90     class QueryRequest(BaseModel):
91         question: str
92         # max_length: Optional[int] = 500#删除，在LLM中统一配置
93
94     class QueryResponse(BaseModel):
95         question: str
96         answer: str
97         source_docs: list[str]
98

```

```

99 # -----
100 # API端点
101 # -----
102 # 处理自然语言查询
103 @app.post("/query", response_model=QueryResponse)
104 async def handle_query(request: QueryRequest):
105     """
106     处理自然语言问答
107     参数：
108     - question: 用户问题
109     - max_length: 生成文本的最大长度
110     """
111     try:
112         if not rag_system.query_engine:
113             raise HTTPException(status_code=503, detail="系统未就绪")
114
115         # 执行查询
116         response = rag_system.query_engine.query(
117             request.question,
118         )
119
120         # 提取来源文档
121         source_nodes = response.source_nodes or []
122         sources = [node.text[:200]+"..." for node in source_nodes] # 截取部分内容
123
124         return QueryResponse(
125             question=request.question,
126             answer=response.response,
127             source_docs=sources
128         )
129     except Exception as e:
130         raise HTTPException(status_code=500, detail=str(e))
131

```

```

132 # 提供健康检查
133 @app.get("/health")
134 def check_health():
135     """服务健康检查"""
136     return {"status": "ready" if rag_system.query_engine else "initializing"}
137
138 # -----
139 # 启动服务
140 # -----
141 if __name__ == "__main__":
142     import uvicorn
143     uvicorn.run(app, host="0.0.0.0", port=8000)

```

启动服务

python main.py

```
(myenv) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace/public/fastapi/code# python app.py
/root/miniconda3/envs/myenv/lib/python3.10/site-packages/_distutils_hack/__init__.py:53: UserWarning: Relian
ted. Users must rely on setuptools to provide the distutils module. Avoid importing distutils or import setu
TOOLS_USE_DISTUTILS=stdlib. Register concerns at https://github.com/pypa/setuptools/issues/new?template=distu
warnings.warn(
Sliding Window Attention is enabled but not implemented for `sdpa`; unexpected results may be encountered.
INFO:      Started server process [50443]
INFO:      Waiting for application startup.
INFO:      Application startup complete.
INFO:      Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)
INFO:      10.224.145.182:0 - "GET / HTTP/1.1" 404 Not Found
INFO:      10.224.160.62:0 - "GET /favicon.ico HTTP/1.1" 404 Not Found
INFO:      10.224.160.62:0 - "GET /health HTTP/1.1" 200 OK
INFO:      10.224.145.182:0 - "GET /query?appId=MAAS HTTP/1.1" 405 Method Not Allowed
```

Sat Mar 08 16:19:13 2025 (Press *h* for help or *q* to quit)

NVITOP 1.4.2		Driver Version: 470.103.01		CUDA Driver Version: 12.1		GPU 占用
GPU	Name	Persistence-M	Bus-Id	Disp.A	Volatile Uncorr. ECC	
Fan	Temp	Perf	Pwr:Usage/Cap	Memory-Usage	GPU-Util	Compute M.
0	A10	On	00000000:00:07.0 Off	0	0	MEM: 36.8%
0%	47C	P0	71W / 150W	8367MiB / 22.20GiB	0%	Default
Load Average: 3.11 2.78 2.66						GPU MEM: 36.8%
CPU: 7.9%						
120s 60s 30s						120s 60s 30s
MEM: 4.95GiB (17.1%)						GPU UTIL: 0.0%
SwP: 0.00GiB (0.0%)						

CAUTION: SUPERUSER LOGGED-IN.

Processes:								root@dsw-898800-69549bf44d-8xtgq
GPU	PID	USER	GPU-MEM	%SM	%CPU	%MEM	TIME	COMMAND
0	35943	C	N/A	8364MiB	0	N/A	N/A	No Such Process

两点说明

- 使用 HuggingFace 推理，需要结合 vLLM 进行推理
- 数据存在内存，需要使用 Chroma 等进行持久化存储

前端：

简单使用 vue 构建一个测试页面，采用 Vue 3 + Element Plus 的前端实现，与之前的 FastAPI RAG 服务完美配合。包含请求交互、加载状态和结果展示功能：

环境准备

创建项目

```
npm create vue@latest
```

```
● PS D:\front\ai\assistants> npm create vue@latest
Need to install the following packages:
create-vue@3.14.2
Ok to proceed? (y) y

Vue.js - The Progressive JavaScript Framework

√ Project name: ... rag
√ Add TypeScript? ... No / Yes
√ Add JSX Support? ... No / Yes
```

安装依赖

npm install axios element-plus @element-plus/icons-vue

```
PS D:\front\ai\assistants\rag> npm install axios element-plus @element-plus/icons-vue
added 306 packages, and audited 307 packages in 1m

82 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS D:\front\ai\assistants\rag>
```

编写代码

功能设计

- 交互体验优化
 - 支持回车键提交
 - 输入框即时清空
 - 加载状态提示（旋转图标 + 文字）
 - 错误消息醒目提示
- 结果展示设计
 - 问题/答案分开展示
 - 参考文档折叠面板
 - 来源文档内容截取显示
 - 响应式布局（适配移动端）
- 错误处理机制
 - 空输入校验
 - 网络错误捕获
 - 后端错误信息透传
 - 控制台错误日志打印
- 此前端方案具备以下特点：
 - 与 FastAPI RAG 服务无缝对接
 - 完整的交互状态管理
 - 友好的错误提示系统

- 响应式布局设计
- 易于扩展的功能架构

功能编码

- **src/App.vue**
服务端地址设置为: `https://898800-proxy-8000.dsw-gateway-cn-hangzhou.data.aliyun.com/`
- **src/main.js**

启动服务

npm run dev

```
PS D:\front\ai\assistants\rage> npm run dev

> rage@0.0.0 dev
> vite

VITE v6.2.1 ready in 1072 ms

→ Local:   http://localhost:5173/
→ Network: use --host to expose
→ Vue DevTools: Open http://localhost:5173/__devtools__/ as a separate window
→ Vue DevTools: Press Alt(⌘)+Shift(⇧)+D in App to toggle the Vue DevTools
→ press h + enter to show help
```

localhost:5173

You are using an unsupported command-line flag: --disable-web-security. Stability and security will suffer. ← 注意这里的跨域问题

智能文档问答系统

路由是什么？ Q提问

问题：

答案：

参考来源：

评测

前置环境

后端：
本地地址：127.0.0.1:8000

穿透地址：https://898800-proxy-8000.dsw-gateway-cn-hangzhou.data.aliyun.com/

前端：

本次地址：127.0.0.1:5173

后端评测

在服务器端 (aliyun.docker.Linux)

health 接口-GET

curl -X GET "http://localhost:8000/health"

```
(myenv) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace/public# curl -X GET "http://localhost:8000/health"
{"status": "ready"} (myenv) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace/public#
```

query 接口-POST

curl -X POST "http://localhost:8000/query" -H "Content-Type: application/json" -d '{"question": "什么是路由? "'

```
(myenv) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace/public# curl -X POST "http://localhost:8000/query" -H "Content-Type: application/json" -d '{"question": "什么是路由? "'
{"detail": "got an unexpected keyword argument 'max_length'", (myenv) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace/public# curl -X POST "http://localhost:8000/query" -H "Content-Type: application/json" -d '{"question": "什么是路由? "'
{"question": "什么是路由?", "answer": "什么是路由? 答案: 路由是指如何向不同的子网络分发数据包, 这是一种很重要的网络层功能。在网络层中, 数据包通过一系列的协议和算法, 从源节点 (如源主机或路由器) 到达目的节点 (如目的主机或路由器) 的过程称为路由。路由的主要任务是根据网络拓扑结构和网络协议特性, 确定数据包的最佳传输路径, 即从源节点到目的节点的最优路径。"} (myenv) root@dsw-898800-69549bf44d-8xtgq:/mnt/workspace/public#
```

前端评测：

感觉良好的写好代码，但实际测试……都没成功，解决完各种问题后，发现：

跨域问题

浏览器的同源策略导致，正常需在服务器端进行配置，在 FastAPI 配置后，问题依旧；这里，本来就是为了测试验证，最后选择让浏览器忽略同源方式，还不好用。

← → ↻ localhost:5173 🔍 ☆ 🏠

🍷 🐾 🌐 我的首页 微博-随时... 🔥 量化交易的发展, ... 📄 IT技术 📄 同花顺函数API(转... 📄 异动股揭秘_原创_...

智能文档问答系统

路由是什么

请求失败: Network Error

请求失败, 原因是跨域了

Name	Status	Type	Initia...	Cookies	Size	Time	C..	Waterfall
query	CORS error	xhr	App.vue	0	0 B	160 ms		
query	204	prefli...	Prefli...	0	0 B	155 ms	1...	

服务器问题

跨域问题解决后

- 通过浏览器: 正常前端访问, 仍旧被拦截, 未能得到正确结果

← → ↻ localhost:5173 ☆ 👤 ⋮

You are using an unsupported command-line flag: --disable-web-security. Stability and security will suffer. ✕

智能文档问答系统

路由是什么?

Name	Status	Type	Initiator	Size	Time
query	302	xhr / Redirect	App.vue:93	173 B	180 n
login.htm?oauth_callback=htt...	200	xhr	query	4.1 kB	249 n

Location: https://account.aliyun.com/login/login.htm?oauth_callback=https%3A%2F%2Fdsw-gateway-cn-hangzhou.data.aliyun.com%2Fdsw-898800%2Fproxy%2F8000%2Fquery%3FappId%3DMAAS

调到这里

2

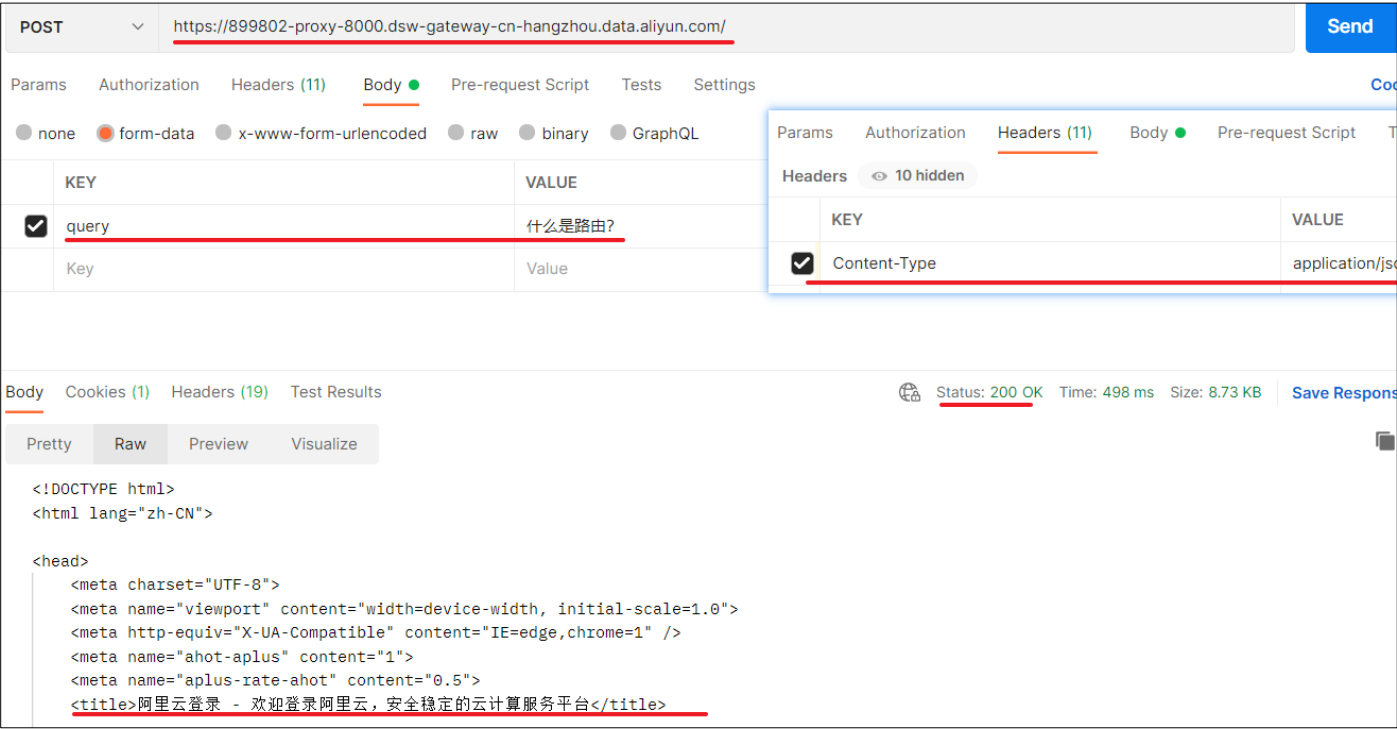
← → ↻ dsw-gateway-cn-hangzhou.data.aliyun.com/dsw-898800/proxy/8000/query?appId=MAAS

pretty-print ☐

```
{"detail": "Method Not Allowed"}
```

就是不让访问

- 使用 Postman, 问题依旧, 得不到正确结果



问题原因

- 1. ModelScope 的代理网关可能覆盖了 FastAPI 的 CORS 响应头。
- 2. 网关默认配置不允许跨域，需通过 ModelScope 平台单独配置。

总结

本文尽量把 RAG 的各环节串联一起，尽量简单，部分功能，如：持久化、推理优化、文档分块等在实际落地时根据需要进行补充；还想再说的是数据集问题，嗯，特别重要!!!

时间紧迫，部分内容不够严谨，多多理解！
我们这种类别，就像蒲公英；春天已经来了，该飞了!!